

周报 2019.4.22~4.28

Done

1.迁移学习小结

- 一些基本概念：

迁移学习

From GitHub: [jindongwang/transferlearning](https://github.com/jindongwang/transferlearning)

简介

迁移学习(transfer learning)通俗来讲，**就是运用已有的知识来学习新的知识，核心是找到已有知识和新知识之间的相似性**，用成语来说就是举一反三。由于直接对目标域从头开始学习成本太高，我们故而转向运用已有的相关知识来辅助尽快地学习新知识。

比如，已经会下中国象棋，就可以类比着来学习国际象棋；已经会编写Java程序，就可以类比着来学习C#；已经学会英语，就可以类比着来学习法语；等等。世间万事万物皆有共性，如何合理地找寻它们之间的相似性，进而利用这个桥梁来帮助学习新知识，是迁移学习的核心问题。

术语

- 已有的知识叫做**源域(source domain)**，要学习的新知识叫**目标域(target domain)**
- 迁移学习按照学习方式划分：
 - 基于**样本**的迁移通过对源域中有标定样本的加利用完成知识迁移；
 - 基于**特征**的迁移通过将源域和目标域映射到相同的空间（或者将其中之一映射到另一个的空间中）并最小化源域和目标域的距离来完成知识迁移；
 - 基于**模型**的迁移将源域和目标域的模型与样本结合起来调整模型的参数；
 - 基于**关系**的迁移则通过在源域中学习概念之间的关系，然后将其类比到目标域中，完成知识的迁移。

相比传统机器学习的优点（对数据要求宽松、可以让相关领域帮忙）

传统机器学习在应对数据的分布、维度，以及模型的输出变化等任务时，模型不够灵活、结果不够好，而迁移学习放松了这些假设。在数据分布、特征维度以及模型输出变化条件下，有机地利用源域中的知识来对目标域更好地建模。另外，在有标定数据缺乏的情况下，迁移学习可以很好地利用相关领域有标定的数据完成数据的标定。

需要注意的风险

理论上，任何领域之间都可以做迁移学习。但是，如果源域和目标域之间相似度不够，迁移结果并不会理想，出现所谓的**负迁移 (negative transfer)** 情况。比如，一个人会骑自行车，就可以类比学电动车；但是如果类比着学开汽车，那就有点天方夜谭了。如何找到相似度尽可能高的源域和目标域，是整个迁移过程最重要的前提。

资料

迁移学习介绍PPT_王晋东_中文: http://jd92.wang/assets/files/l08_tl_zh.pdf

迁移学习中的领域自适应方法: http://jd92.wang/assets/files/l12_da.pdf

深度迁移报告_龙明盛: <http://ise.thss.tsinghua.edu.cn/~mlong/doc/deep-transfer-learning-talk.pdf>

迁移学习简明手册_王晋东:

http://jd92.wang/assets/files/transfer_learning_tutorial_wjd.pdf

2.ChinaVis 2019 挑战赛 项目

- 学习React:

React.js 小书

<http://huziketang.mangojuice.top/books/react/>

第一阶段

简介

React.js是用于构建页面UI的库，将功能抽象成组件，和数据进行交互

前端组件化

- 直接写出HTML结构和JS事件（但是难以复用）
- class, render()方法返回一个HTML字符串（但是字符串加不了DOM事件的function）
 - 通过createDOMFromString把字符串变成DOM元素
 - 通过addEventListener添加DOM事件
 - 通过appendChild操作DOM元素
- 生成复用的组件: class, 有构造函数, state中有数据成员, 有函数成员可以绑定在交互事件上, 有render方法返回携带各种函数功能的DOM元素

优化DOM操作

- 通过全局变量state的变化, 直接修改render里面的HTML字符串, 来更新DOM结构
- 通过this.el, 每次state变化的时候, 插入newEl, 删除oldEl
- 这样就不用手动管理DOM, 可以自动渲染组件

抽象公共组件类

- 父类, 包含构造函数constructor、setState、renderDOM
- 子类继承父类, 实现render, 添加属性props

JSX

- JSX以HTML的语法在JS里写, 方便; 被编译成JavaScript对象
- 然后ReactDOM.render将其变成DOM元素/或者如ReactNative变成APP
- 优点是处理JS对象来更新组件比处理DOM快很多

render方法

- Render () { return (JSX对象) }
- 使用JavaScript的表达式 var a = 1, 使用{ a }嵌入JSX中
- Class -> className ; for -> htmlfor

组合和嵌套

- Class Title extends Component 使用<Title> </Title>组合
- 自定义的组件都必须要用大写字母开头，普通的 HTML 标签都用小写字母开头。

事件监听

- 封装了一系列on*的事件，<h1 onClick={this.function}>233</h1>，现在只能用在普通的组件上，不能用在自定义的组件上
- 事件监听函数传入event对象，是通用的，如属性e.target.innerHTML
- 调用的事件函数不是使用this对象方法，使用this.function.bind(this, EleList)来绑定this并向事件函数传入元素

State

- 状态中存储全局变量，调用：this.state.element
- setState方法执行后将自动更新state，重新调用render方法，重新渲染
- 注意这是缓存延后执行的，想要连续使用setState，必须传入函数操作，不然会合并处理

Props

- 组件中的标签属性作为props对象的键值 <button ele={ {t1:'a', t2:'b'} }></button>
 - 使用const ele = this.props.ele
 - 可以直接传入一个匿名函数，也可以传入类内的其他函数onClick = {this.myClick}（这是对于自定义的组件，onClick指自定义的属性，而不是默认的click触发事件）
 - 使用defaultProps设置默认配置属性
 - 传入props后，组件内部不能改变props，但是可以通过父组件传入新的props
- 讨论、绘制时间-地点的人物轨迹图

小结

迁移学习是联邦学习的基础，先看了一些；参加ChinaVis挑战赛，开始学习ReactJs的使用，并承担一部分视图的实现。

下周五一节，我想继续学习一会。

Plan

短期计划

1. 看联邦学习、迁移学习的实现方法和样例
2. 继续学习react相关知识，实现ChinaVis2019的前端的一个视图

中期计划

1. 写本科毕业设计
2. 写组会报告

长期计划

1. 6月份时，完成上述几个事项